FIG.1

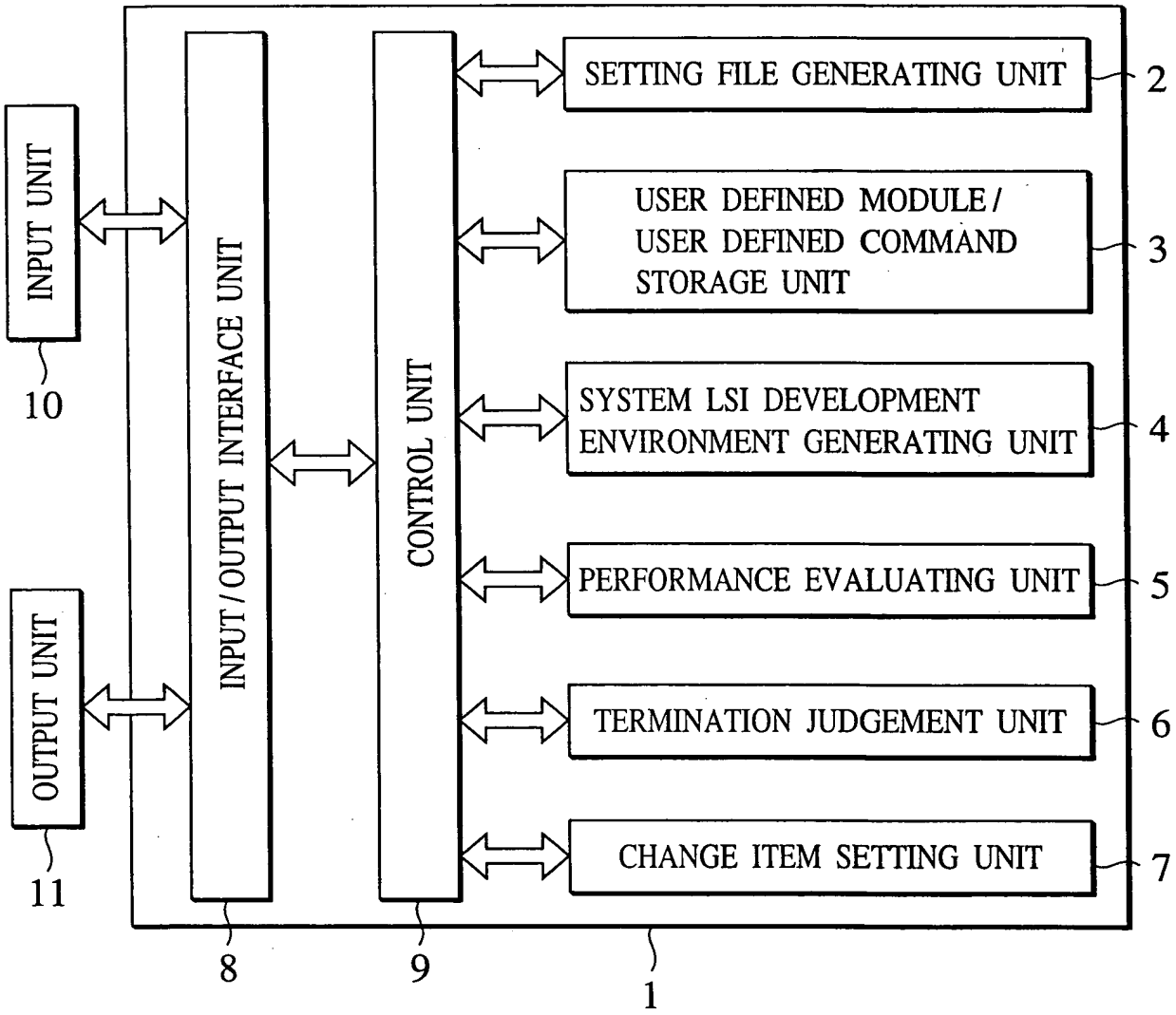
INPUT UNIT
10
OUTPUT UNIT
11
INPUT/OUTPUT INTERFACE UNIT
8
CONTROL UNIT
9
SETTING FILE GENERATING UNIT
2
USER DEFINED MODULE/
USER DEFINED COMMAND
STORAGE UNIT
3
SYSTEM LSI DEVELOPMENT
ENVIRONMENT GENERATING UNIT
4
PERFORMANCE EVALUATING UNIT
5
TERMINATION JUDGEMENT UNIT
6
CHANGE ITEM SETTING UNIT
7
1

FIG.2

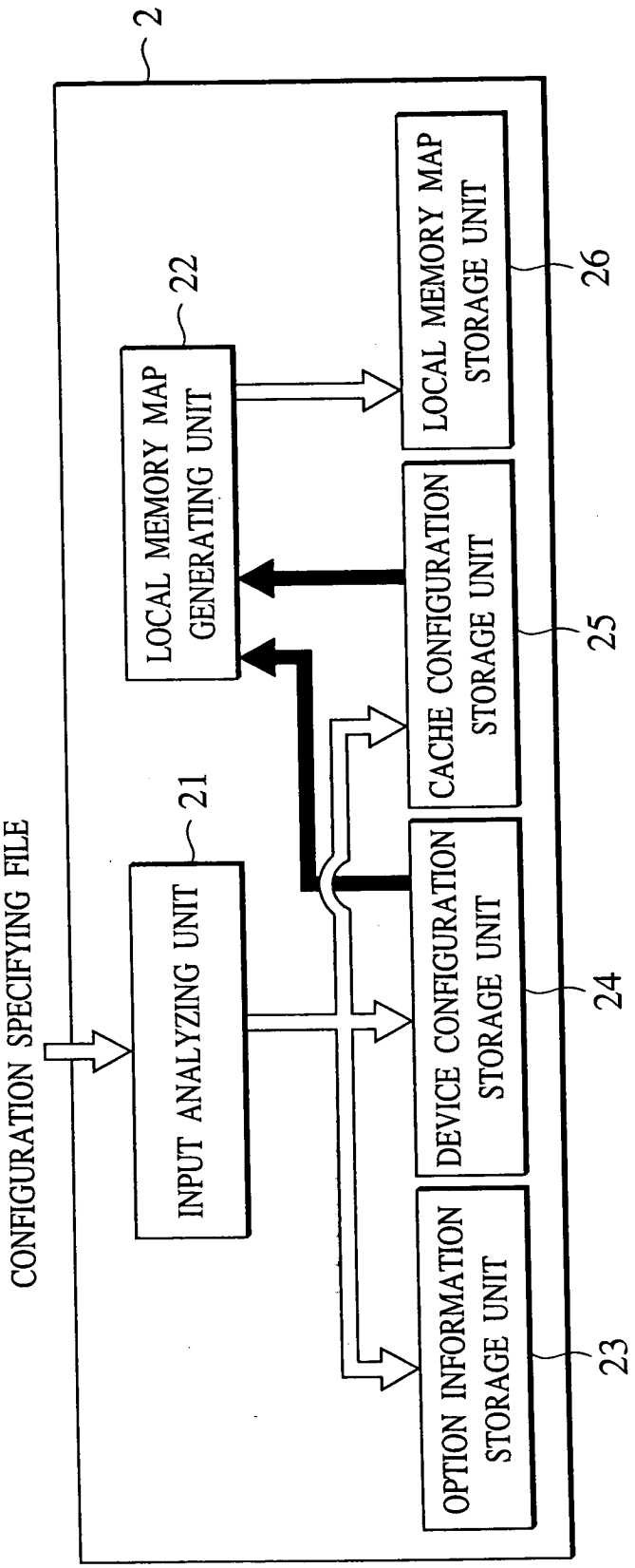
2
CONFIGURATION SPECIFYING FILE
INPUT ANALYZING UNIT
21
LOCAL MEMORY MAP GENERATING UNIT
22
OPTION INFORMATION STORAGE UNIT
23
DEVICE CONFIGURATION STORAGE UNIT
24
CACHE CONFIGURATION STORAGE UNIT
25
LOCAL MEMORY MAP STORAGE UNIT
26

FIG.3

```
//configuration data
CHIP_NAME="SimpleChip1"://CHIP NAME
FREQUENCY=200:          //FREQUENCY UNIT.MHz,
//PROCESSOR1
P_MODULE[Processor1](
        P_ENGINE(
            //SPECIFICATION OF PROCESSOR CORE
            CORE{
                ID=0; //CORE ID
                //OPTION COMMAND
                DIV=ON; //DIVIDING COMMAND
                MINMAX=ON; //MAXIMUM / MINIMUM VALUE COMMAND
                BIT=ON; //BIT OPERATION COMMAND
            };
            IMEM{
                SIZE=4 ;    //KB
            };
            DMEM{
                SIZE=2 ;    //KB

            BIU{ //BUS INTERFACE COMMAND
                BUS_SIZE=64 ;
            };
            INTC{  //INTERRUPT CONTROLLER
                CHANNEL_BITW=16 ; //NOT DEFINED
                LEVEL=15 ; //NOT DEFINED
            };
            DSU{ //DEBUG UNIT
                INST_ADDR_BREAK_CHANNEL=1 ;
                DATA_ADDR_BREAK_CHANNEL=1;
            };
        };
};
//PROCESSOR2
P_MODULE[Processor2](
        P_ENGINE(
            CORE{
                ID=1; //CORE ID
                //OPTION COMMAND
                DIV=ON; //DIVIDING COMMAND
                MINMAX=OFF; //MAXIMUM / MINIMUM VALUE COMMAND
                BIT=OFF; //BIT OPERATION COMMAND
            };
            IMEM{ //COMMAND RAM
                SIZE=8 ;    //KB
            };
            DMEM{ //DATA RAM
                SIZE=20 ;    //KB
            };
            ICACHA{ //COMMAND CACHE
                SIZE=4 ;    //KB
            };
            DCACHE{ //DATA CACHE
                SIZE=8 ;    //KB
            };
        };
        //COPROCESSOR
        COPRO[Cop1]{
            IS_VLIW=YES ; //VLIW TYPE COPROCESSOR
            VLIW_BTW=32 ; //COMMAND LENGTH 32 BITS
            ISA_DEFINE="cop1.isa" ; //ISA DEFINITION, ANOTHER FILE
            RTL    ="cop1.v"        //RTL DESCRIPTION, ANOTHER FILE
        };
//USER DEFINITION
UCI{
        ISA_DEFINE="uxi1.uci" ; //ISA DEFINITION, ANOTHER FILE
        RTL="uci1.v" ; //RTL DESCRIPTION, ANOTHER FILE
        SIM="ucimode1.c" ;        //HARDWARE C MODEL,ANOTHER FILE
        };
};
//GLOBAL MAP
GLOBAL_MAP=:schip1.map ; //SPECIFY ANOTHER FILE
//end of file
```

FIG.4A

```
//start : size : name : cache(opt) : shadow_original_start(opt) : scope : type : read_write(opt)
0X0000_0000 : 0X0020_0000 : RAM1 :          : : global : memory : ro ;
0X0080_0000 : 0X0080_0000 : RAM2 : Cache : : global : memory : rw ;
0X8080_0000 : 0X0080_0000 : RAM3 :          : : global : memory : rw ;
```

FIG.4B

```
:RAM1     :0X00000000 : 0X00200000:        :global : ro::     memory:
:Imem0    :0X00200000 : 0X00002000:        : local : rw::      Imem:    in mm
:Imem1    :0X00202000 : 0X00002000:        : local : rw::      Imem:    in mm
:dmem0    :0X00208000 : 0X00004000:        : local : rw::      dmem:    in mm
:dmem1    :0X0020c000 : 0X00004000:        : local : rw::      dmem:    in mm
:icahe_dat :0X00300000 : 0X00004000:       : local : rw:: icache_data:  in mm
:icache_tag:0X00310000 : 0X00004000:       : local : rw:: Icache_tag:   in mm
:RAM2     :0X00800000 : 0X00800000: cache :global : rw::    memory:
:RAM3     :0X00808000 : 0X00800000:        :global : rw::    memory:
```

FIG.4C

```
DECLARING COMMAND MNEMONIC,OPERATION CODE, AUGMENT
void xor(int_reg_modify,int_reg_arc) ;
{
        code16="0000_0010_0000" ;
}
short xori(int_reg_src,signed char_imm);
{
        code16="0000_0011_iiii_iiii";
}
```

FIG.5

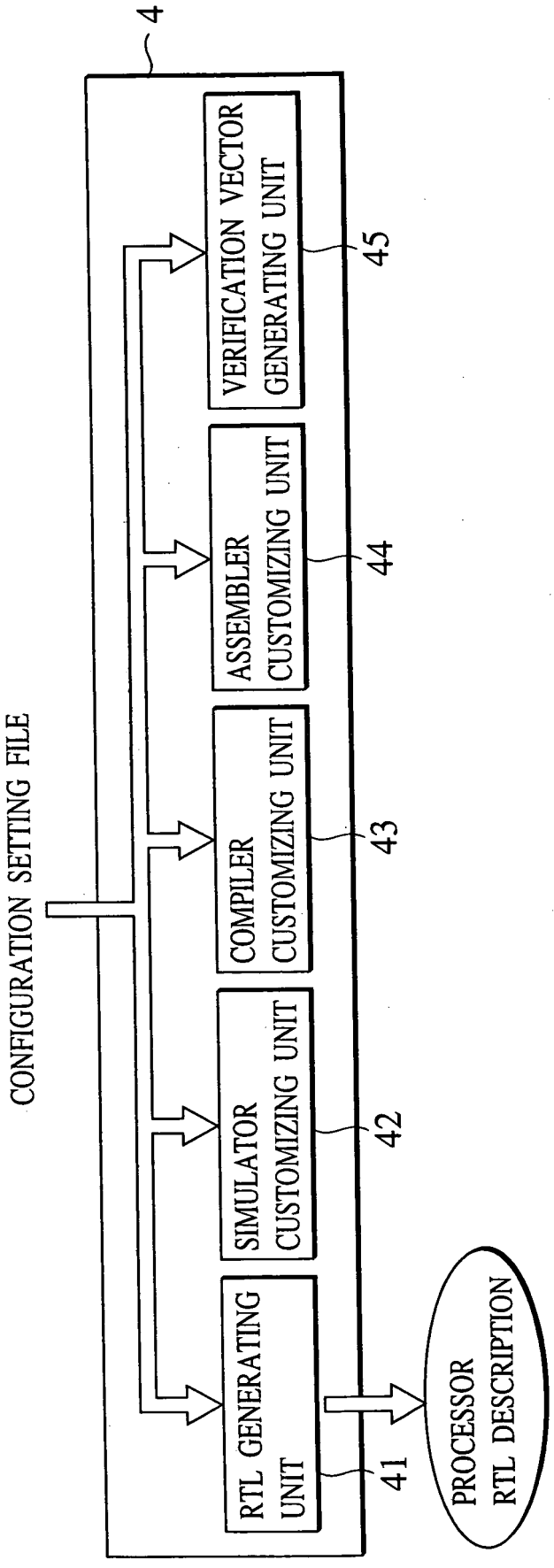
4
CONFIGURATION SETTING FILE
RTL GENERATING UNIT
41
SIMULATOR CUSTOMIZING UNIT
42
COMPILER CUSTOMIZING UNIT
43
ASSEMBLER CUSTOMIZING UNIT
44
VERIFICATION VECTOR GENERATING UNIT
45
PROCESSOR RTL DESCRIPTION

FIG.6

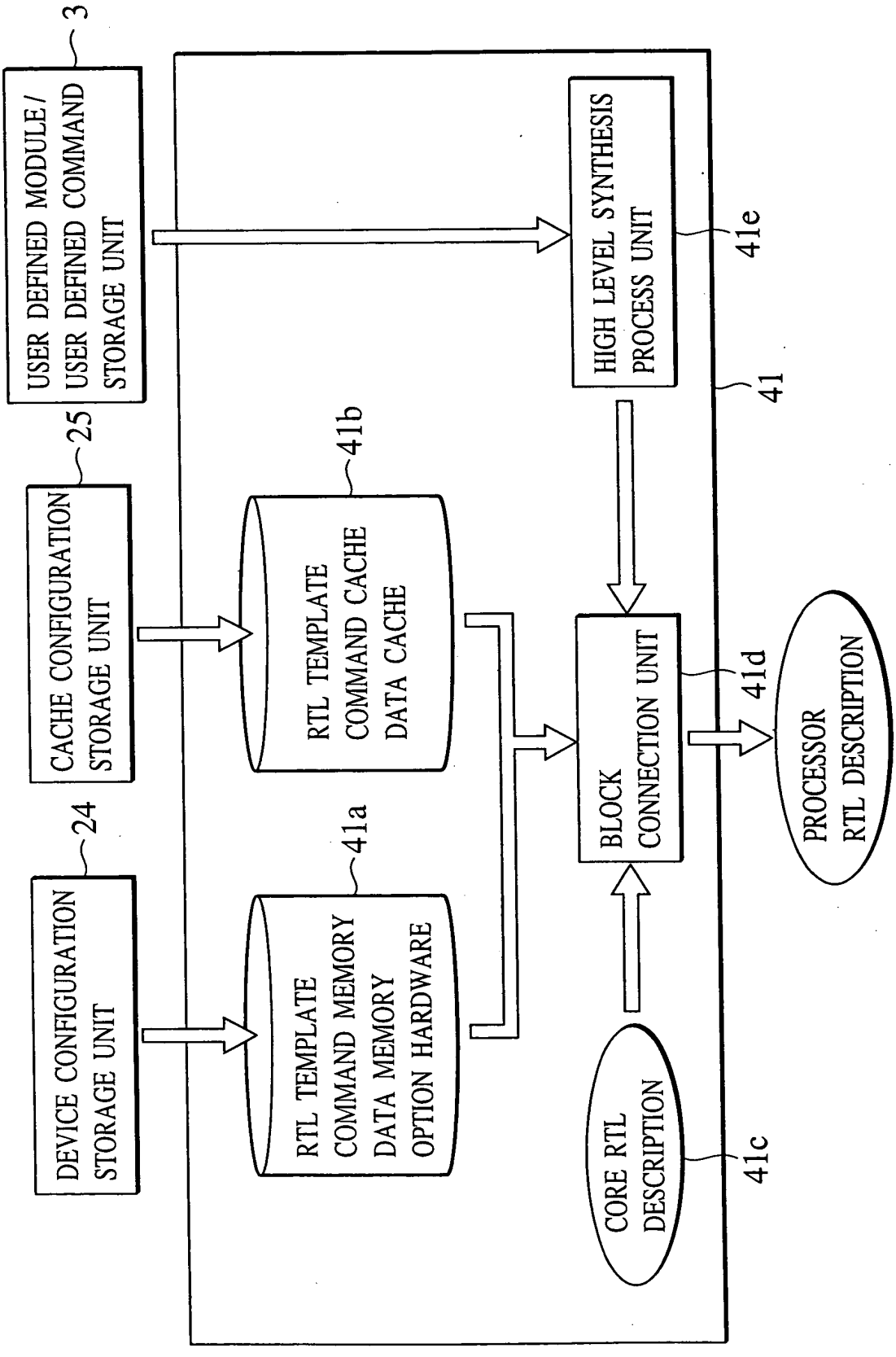
USER DEFINED MODULE / USER DEFINED COMMAND STORAGE UNIT
3
HIGH LEVEL SYNTHESIS PROCESS UNIT
41e
41
CACHE CONFIGURATION STORAGE UNIT
25
RTL TEMPLATE COMMAND CACHE DATA CACHE
41b
DEVICE CONFIGURATION STORAGE UNIT
24
RTL TEMPLATE COMMAND MEMORY DATA MEMORY OPTION HARDWARE
41a
BLOCK CONNECTION UNIT
41d
PROCESSOR RTL DESCRIPTION
CORE RTL DESCRIPTION
41c

FIG.7

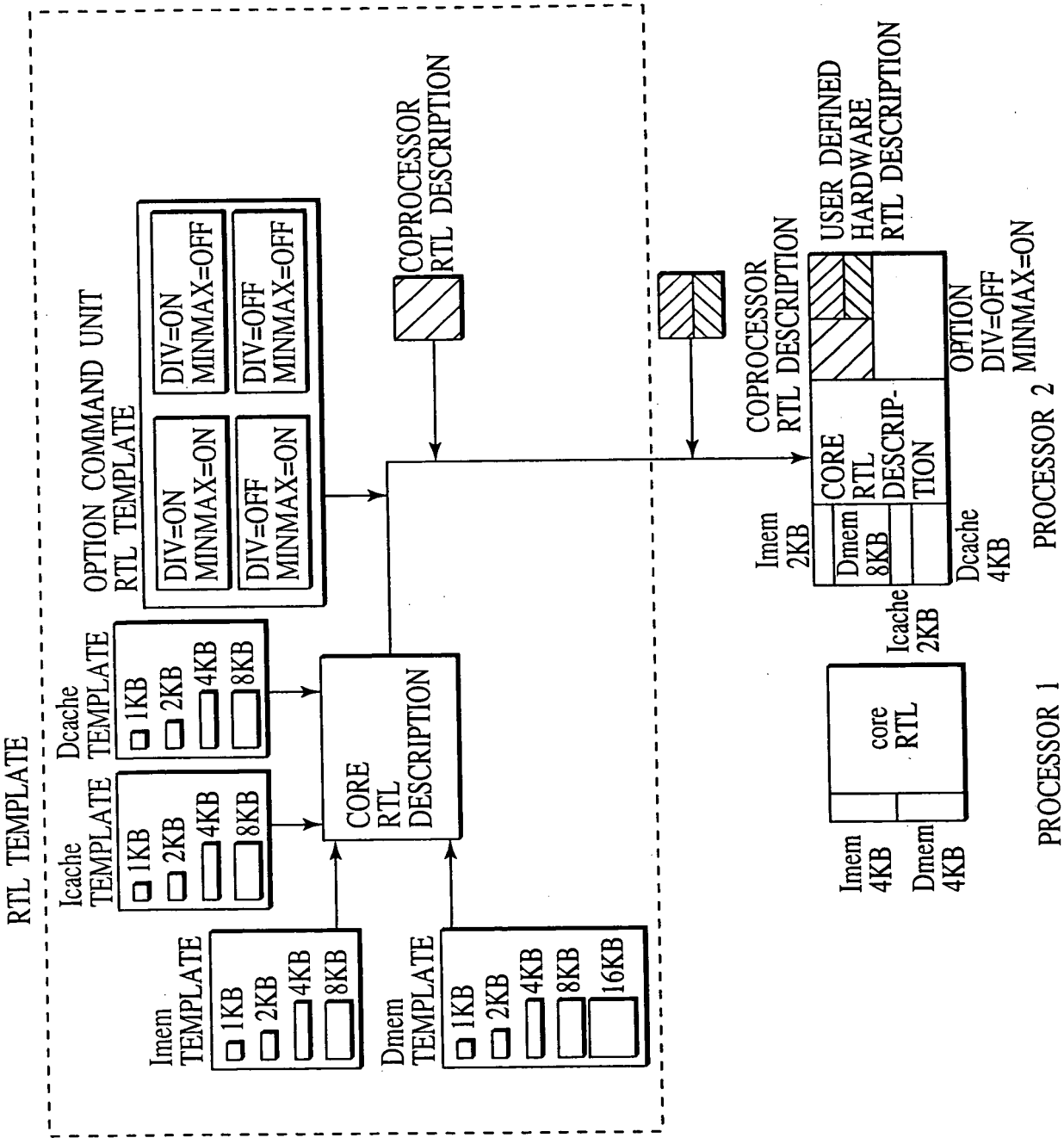
RTL TEMPLATE
Icache TEMPLATE
1KB
2KB
4KB
8KB
Dcache TEMPLATE
1KB
2KB
4KB
8KB
OPTION COMMAND UNIT RTL TEMPLATE
DIV=ON MINMAX=ON
DIV=OFF MINMAX=ON
DIV=ON MINMAX=OFF
DIV=OFF MINMAX=OFF
Imem TEMPLATE
1KB
2KB
4KB
8KB
Dmem TEMPLATE
1KB
2KB
4KB
8KB
16KB
CORE RTL DESCRIPTION
COPROCESSOR RTL DESCRIPTION
COPROCESSOR RTL DESCRIPTION
USER DEFINED HARDWARE RTL DESCRIPTION
Imem 2KB
CORE RTL DESCRIP-TION
Dmem 8KB
Dcache 4KB
OPTION DIV=OFF MINMAX=ON
PROCESSOR 2
Icache 2KB
core RTL
Imem 4KB
Dmem 4KB
PROCESSOR 1

FIG.8

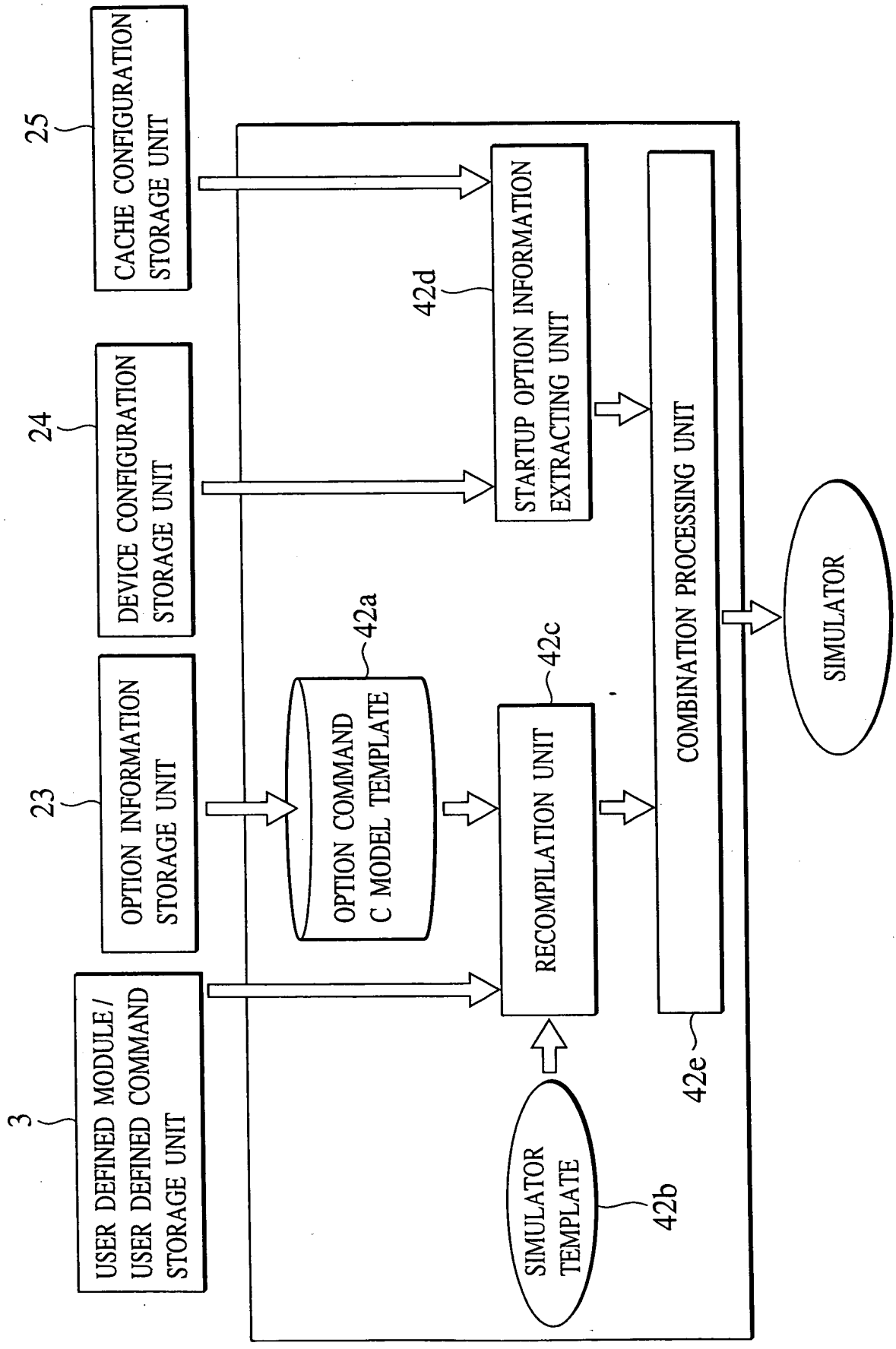
3
USER DEFINED MODULE / USER DEFINED COMMAND STORAGE UNIT
23
OPTION INFORMATION STORAGE UNIT
24
DEVICE CONFIGURATION STORAGE UNIT
25
CACHE CONFIGURATION STORAGE UNIT
42a
OPTION COMMAND C MODEL TEMPLATE
42b
SIMULATOR TEMPLATE
42c
RECOMPILATION UNIT
42d
STARTUP OPTION INFORMATION EXTRACTING UNIT
42e
COMBINATION PROCESSING UNIT
SIMULATOR

FIG.9A

```
-opt_minmax=ON
-opt_div=ON
-opt_bit=OFF
-rom=0x00000000,0x00200000
-imem=0x00200000,0x00300800
-dmem=0x00300000,0x00400000
-icache=0x00400000,0x007f0000
-dcache=0x007f0000,0x00800000
-ram=0x00800000,0x01000000
-ram-shadow=0x80800000,0x81000000
:
```

FIG.9B

```
-rom=0x00000000,0x00200000
-imem=0x00200000,0x00300800
-dmem=0x00300000,0x00400000
-icache=0x00400000,0x007f0000
-dcache=0x007f0000,0x00800000
-ram=0x00800000,0x01000000
-ram-shadow=0x80800000,0x81000000
:
```

FIG.9C

```
// OPTION COMMAND
void_asm min(int_reg_modify,int_reg_src);
void_asm max(int_reg_modify,int_reg_src);
// USER DEFINED COMMAND
void_asm xor(int_reg_modify,int_reg_src);
short_asm xori(int_reg_src,signed char_imm);
:
```

FIG.10

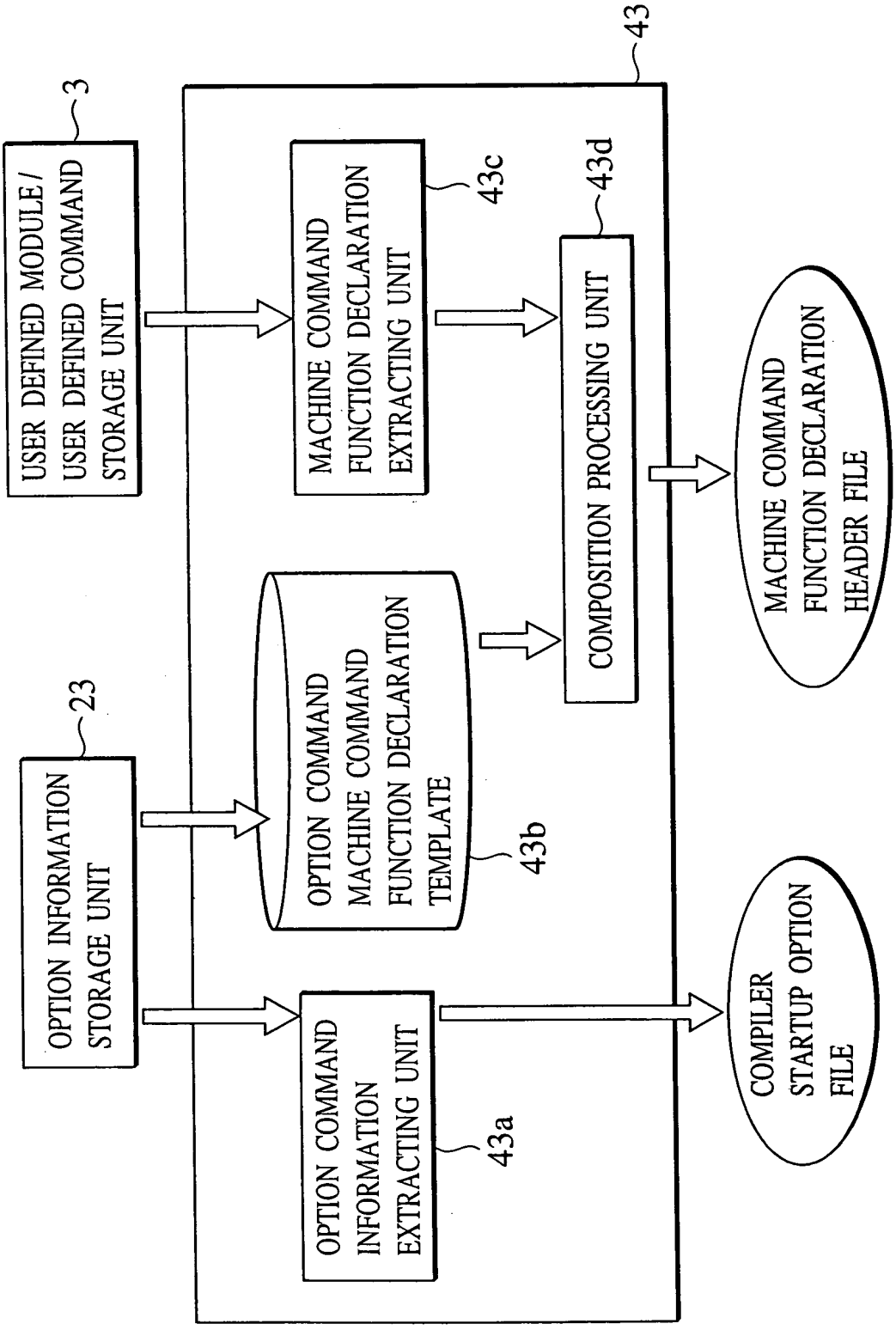
43
3
USER DEFINED MODULE/
USER DEFINED COMMAND
STORAGE UNIT
MACHINE COMMAND
FUNCTION DECLARATION
EXTRACTING UNIT
43c
43d
COMPOSITION PROCESSING UNIT
MACHINE COMMAND
FUNCTION DECLARATION
HEADER FILE
23
OPTION INFORMATION
STORAGE UNIT
OPTION COMMAND
MACHINE COMMAND
FUNCTION DECLARATION
TEMPLATE
43b
OPTION COMMAND
INFORMATION
EXTRACTING UNIT
43a
COMPILER
STARTUP OPTION
FILE

FIG.11A

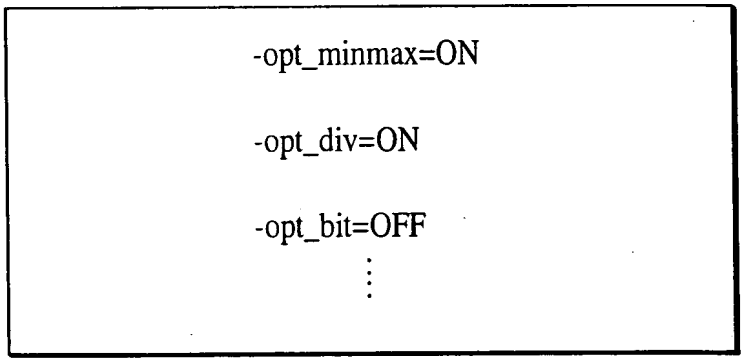

FIG.11B

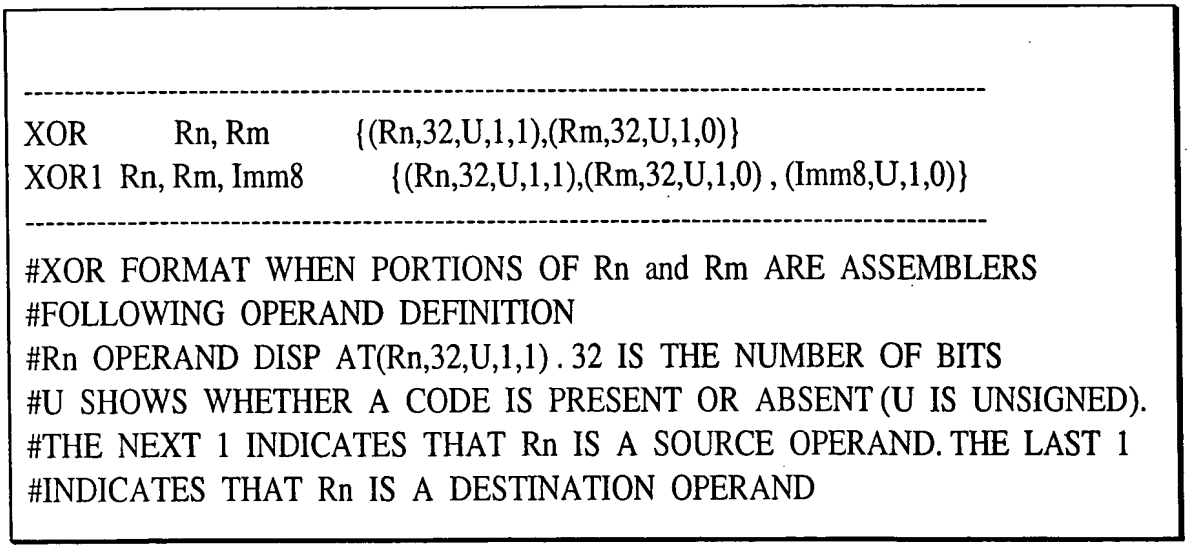

FIG.12A

```
int x=0 ;

main( ) {

        int i , j ;
        for(i=0 ; i<10 ; i++)
              for(j=0 ; j<5 ; j++)
                   x+=i * j ;
        printf("result=%d¥n", x) ;
        }
```

FIG.12B

```
int x=0 ;

main( ) {

        int i , j ;
        for(i=0 ; i<10 ; i++){
_START(1)
              for(j=0 ; j<5 ; j++)
                   x+=i * j ;
_END(1)
              }
        printf("result=%d¥n", x) ;
        }
```

FIG.13A

NUMBER	START ADDRESS	END ADDRESS	NUMBER OF COMMANDS
1	0x8001e	0x80038	370

FIG.13B

NUMBER	START ADDRESS	END ADDRESS	NUMBER OF CYCLES
1	0x8001e	0x80038	500

FIG.13C

CACHE SIZE	CACHE ERROR RATE
1Kbytes	0.191
2Kbytes	0.148
4Kbytes	0.109
8Kbytes	0.087
16Kbytes	0.066

FIG.14

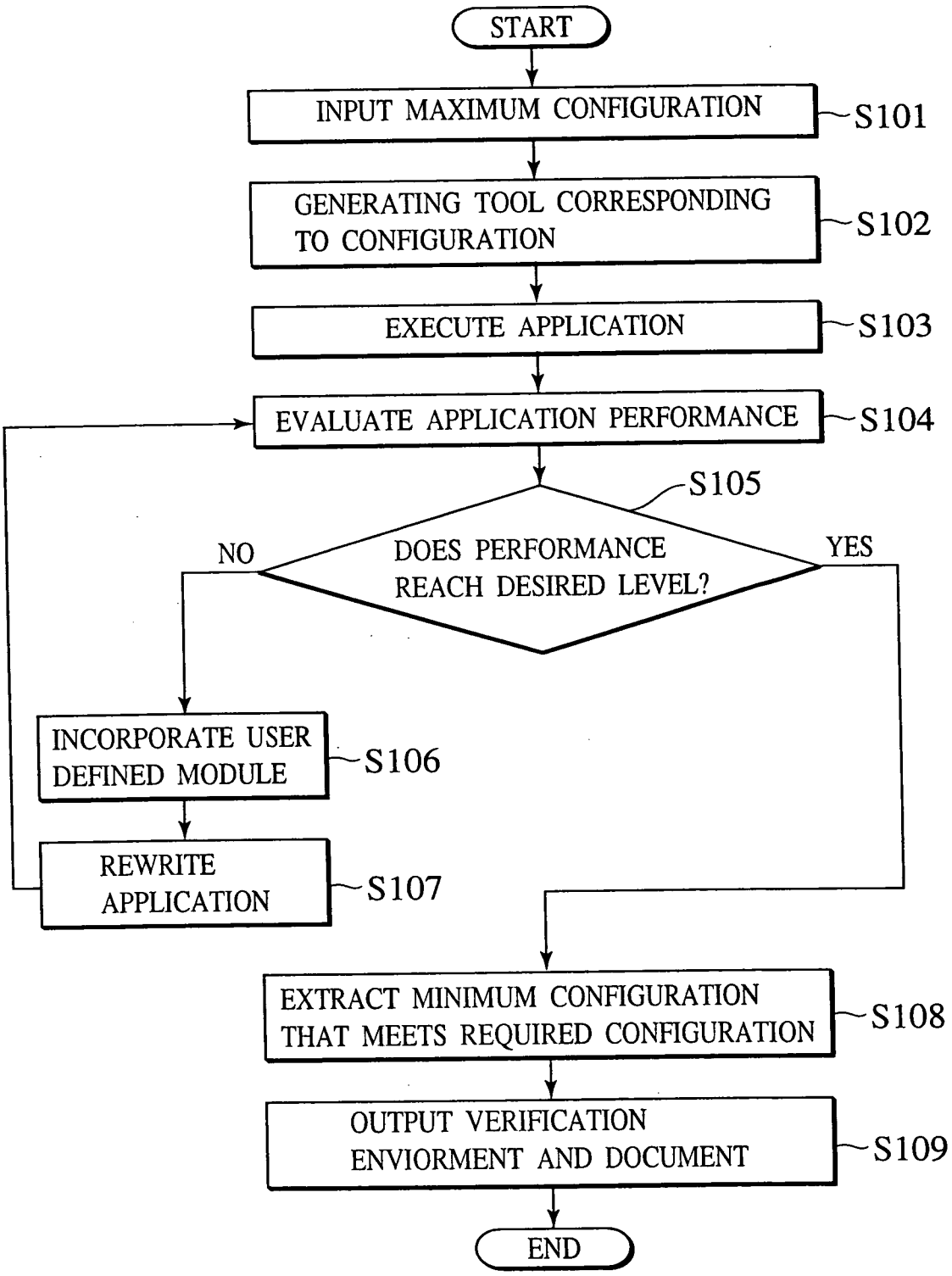
START
INPUT MAXIMUM CONFIGURATION
S101
GENERATING TOOL CORRESPONDING TO CONFIGURATION
S102
EXECUTE APPLICATION
S103
EVALUATE APPLICATION PERFORMANCE
S104
S105
DOES PERFORMANCE REACH DESIRED LEVEL?
NO
YES
INCORPORATE USER DEFINED MODULE
S106
REWRITE APPLICATION
S107
EXTRACT MINIMUM CONFIGURATION THAT MEETS REQUIRED CONFIGURATION
S108
OUTPUT VERIFICATION ENVIORMENT AND DOCUMENT
S109
END

FIG.15A

```
        move R5, 0x00080000
        move R6, 0x00090000
        move R7, 0x00080100
    loop :
        ld          R1,   (R5)
        ld          R2,  4(R5)
        call        _Culc          // TO SUBROUTINE
        st          R1,   (R6)
        add         R6,     4
        add         R5,     8
        not_equal       R5,      R7, loop //if R5 !=R7 then jump roop
```

FIG.15B

```
Move R1, 0x00080000        //SUBSTITUTE START ADDRESS INTO R1
Move R2, 0x00080100        //SUBSTITUTE START ADDRESS INTO R2
Move R3, 0x00090000        //SUBSTITUTE COMPUTATION RESULT STRAGE START LOCATION INTO R3
Move R4, 0x00000001        //SUBSTITUTE DSP START VALUE INTO R4
St_cntibus R1, (cntibus_addr1) //SPECIFY READ START LOCATION
St_cntibus R2, (cntibus_addr2) //SPECIFY READ END LOCATION
St_cntibus R3, (cntibus_addr3) //SPECIFY START LOCATION OF REWRITING COMPUTATION RESULT
St_cntibus R4, (cntibus_addr4) //START DSP PROCESSING
(COREPROCESSOR CONTINUES OTHER PROCESSING, WAIT FOR THE END OF DSP,AND CONTINUES PROCESSING)
```

FIG.15C

```
Tyedef unsigned long     address;
class DSP_HWEngine : public =HWEngine, public ControlBus
{
public :
   virtual bool read_cntibus(address adr, unsigned long* data) ;
   virtual bool write_cntibus(address adr, unsigned long data) ;
   virtual bool do_command(unsigned long s1,unsigned long s2,unsigned long operand,
                           unsined long* ret) ;
}
```

FIG.16

```
Bool DSP_HWEngine ::read_cntibus(address adr, unsigned loog* data)
{
        * date=contibus[adr];
        return true ;
}
bool DSP_HWEngine ::write_cntibus(address adr, unsigned loog data)
{
        if(adr==cntibus_addr4)//VALUE IS WRITTEN IN DSP CONTROL REGISTER
        {
                PROCESSING IS PERFORMED FOR WRITEN VALUE
                return true ;
        }
    contibus[adr]=data
    return true ;
}

bool DSP_HWEngine :: do_command(unsigned long s1, unsigned long s2, unsigned long operand,
unsigned long* ret)
{
if(operand&0x0000ff00)==0x60)
{
   PROCESSING OF COMMAND CODE 0x60 IS WRITTEN, COMMAND SOURCE IS
   DO_COMMAND FUNCTION AUGMENT, A1 AND A2, DESTINATION IS "RET"
   THE COMPUTATION RESULT IS PLACED IN "RET", AND IS RETURNED
   *RET=RESULT
   return true ;
   }
WHEN A USER DEFINED COMMAND IS PRESENT,"IF" ARE ARRANGED IN PLURALITY
Else if( )
{


    }
}
```

FIG.17A

```
Class CORE
{
public :
     void one_step_execute() ;
};
```

FIG.17B

```
Core1_config_set()
{
        //CORE1 OBJECT GENERATION
        //CORE1 OPTION COMMAND SETTING
        //SETTING OF HARDWARE ENGINE AND THE LIKE
        //OTHER SETTINGS(CACHE OR COPROCESSOR)
}
Core2_config_set()
{
        //CORE2 OBJECT GENERATION
        //CORE2 OPTION COMMAND SETTING
        //SETTING OF HARDWARE ENGINE AND THE LIKE
        //OTHER SETTINGS(CACHE OR COPROCESSOR)
}
Core3_config_set()
{
        //CORE3 OBJECT GENERATION
        //CORE3 OPTION COMMAND SETTING
        //SETTING OF HARDWARE ENGINE AND THE LIKE
        //OTHER SETTINGS(CACHE OR COPROCESSOR)
}
```

FIG.18A

```
Module_set()
{
  //CALL SET FUNCTION OF EACH CORE AND GENERATE AND SET OBJECT
  Core1_config_set() ;
  Core2_config_set() ;
  Core3_config_set() ;
}
```

FIG.18B

```
One_set()
{
    Core1>one_step_execute() ;
    Core2>one_step_execute() ;
    Core3>one_step_execute() ;
}
```

FIG.19